UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/817,880	03/26/2001	Sean E. Trowbridge	MS167381.1	7613

27195 7590 07/18/2008

AMIN. TUROCY & CALVIN, LLP
24TH FLOOR, NATIONAL CITY CENTER
1900 EAST NINTH STREET
CLEVELAND, OH 44114

EXAMINER	
RUTTEN, JAMES D	
ART UNIT	PAPER NUMBER
2192	

NOTIFICATION DATE	DELIVERY MODE
07/18/2008	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

docket1@thepatentattorneys.com
hholmes@thepatentattorneys.com
lpasterchek@thepatentattorneys.com

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 09/817,880
Filing Date: March 26, 2001
Appellant(s): TROWBRIDGE, SEAN E.

Himanshu S. Amin, Reg. No. 40,894
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed 4/26/07 appealing from the Office action mailed 10/16/06. Further, this is a revised Examiner's Answer containing changes from the 7/31/07 Examiner's Answer in sections (6)-(8) below in response to the 6/16/08 Order Returning Undocketed Appeal To Examiner.

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is substantially correct. The changes are as follows (note that the changes that were not present in

the 7/31/07 Examiner's Answer as pointed out in the 6/16/08 Order Returning Undocketed Appeal To Examiner are underlined for clarity):

Claims 3 and 4 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent Number 5,761,512 to Breslau et al. (hereinafter "Breslau") in view of U.S. Patent Number 6,571,389 to Spyker et al. (hereinafter "Spyker"), in view of "HotSpot: A new breed of virtual machine" by Armstrong (hereinafter "Armstrong"), and further in view of U.S. Patent Number 6,126,330 to Knight (hereinafter "Knight") as applied in the rejection of claims 1, 2, 5-10, 12-17, 19, 30, and 31, and further in view of U.S. Patent Number 6,721,946 to Fogarty et al. (hereinafter "Fogarty").

Claim 18 is rejected under 35 U.S.C. 103(a) as being unpatentable over Breslau, Spyker, Armstrong, and Knight as applied in the rejection of claims 1, 2, 5-10, 12-17, 19, 30, and 31, and further in view of prior art of record U.S. Patent Number 6,253,368 to Nelin et al. (hereinafter "Nelin").

GROUNDS OF REJECTION NOT ON REVIEW

The following grounds of rejection have not been withdrawn by the examiner, but they are not under review on appeal because they have not been presented for review in the appellant's brief. Claims 19, 30, and 31 are rejected under 35 U.S.C. § 101 because the claimed invention is directed to non-statutory subject matter.

(7) Claims Appendix

A substantially correct copy of appealed claims 1-33 appears on pages 15- 20 of the Appendix to the appellant's brief. In response to the 6/16/08 Order Returning Undocketed Appeal To Examiner, the minor errors are as follows: Claim 33 on page 20 of appellant's brief contains language which was removed in the 12/15/06 amendment. Recitation of "virtual software" in line 1 should be removed. The following is a clean version of the proper claim 33 as found in the 12/15/06 claim amendment, and has been treated accordingly:

33. A computer implemented system, comprising the following computer executable components:

an execution engine that processes an Intermediate Language (IL) image, the execution engine generating operating environment data associated with a particular user while processing the IL image;

a specialized executable image generated at least in part from the operating environment data, the operating environment data includes at least a set of information to create a specialized executable image according to the particular user, the specialized executable image stored in a repository of one or more other specialized executable images; and

wherein the execution engine selects at least one specialized executable image from the repository if the at least one specialized image matches present operating environment data.

(8) Evidence Relied Upon

Note that prior art of record US 6,456,122 to Remezani has been added in response to the 6/16/08 Order Returning Undocketed Appeal To Examiner. The evidence relied upon is as follows:

5,761,512	BRESLAU et al.	6-1998
6,571,389	SPYKER et al.	5-2003
6,721,946	FOGARTY et al.	4-2004
6,253,368	NELIN et al.	6-2001
6,158,049	GOODWIN et al.	12-2000
6,126,330	KNIGHT	10-2000
6,457,122	REMEZANI	9-2002

Armstrong, "HotSpot: A new breed of virtual machine" JavaWorld <http://www.javaworld.com>, March 1998, 11 pages

Aho et al. "Compilers: Principles, Techniques, and Tools" Addison-Wesley, 1986, Chapter 1

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

- Claims 19, 30 and 31 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

Claim 19 is directed to a "computer-readable medium". However, this medium is not limited to tangible embodiments. A description of computer readable media can be found on page 12, line 21 – page 13, line 5. While this description includes tangible embodiments such as a hard disk, magnetic disk, and CD, page 13 lines 1-5 expressly envisions media to include "other types of media readable by a computer". However, wireless media could be read by a computer, but are considered to be a form of electro-magnetic signal, which appears to be non-statutory. Claims 30 and 31 are drawn to a "signal". However, as seen in the limitations of claim 31, a "signal" could be interpreted to include wireless, or electromagnetic signals which appear to be nonstatutory. Claims that recite nothing but the physical characteristics of a form of energy, such as a frequency, voltage, or the strength of a magnetic field, define energy or magnetism, per se, and as such are nonstatutory natural phenomena. O'Reilly, 56 U.S. (15 How.) at 112-14. Moreover, it does not appear that a claim reciting a signal encoded with functional descriptive material falls within any of the categories of patentable subject matter set forth in § 101. For further information, see Official Gazette, Nov. 22, 2005, 1300 OG 142, "Interim Guidelines for Examination of Patent Applications for Patent Subject Matter Eligibility", Annex IV(c), which can be found online at <http://www.uspto.gov/web/offices/com/sol/og/2005/week47/patgupa.htm>.

- Claims 1, 2, 5-17, 19, 30, and 31 are rejected under 35 U.S.C. 103(a) as being unpatentable over prior art of record U.S. Patent Number 5,761,512 to Breslau et al. (hereinafter "Breslau") in view of prior art of record U.S. Patent Number 6,571,389 to Spyker et al. (hereinafter "Spyker"), in view of "HotSpot: A new breed of virtual machine" by Armstrong

(hereinafter "Armstrong"), and further in view of U.S. Patent Number 6,126,330 to Knight (hereinafter "Knight").

In regard to Claim 1, Breslau teaches *a log to store... information relating to an operating environment of a system* (Figure 2)*, the logged information is employed as feedback to generate a native executable* (Figure 3, item 59). Also see column 3 lines 10-14:

> The techniques of the present invention enable the **automatic compilation** of objects defined by classes for execution environments that satisfy requirements of the classes. Each class has its own **affinity (or affinities) that specifies execution environment characteristics** desired by the class. [emphasis added]

Breslau also discloses that any applicable characteristic can be stored in a log. See column 4 lines 60-64:

> Any characteristic of an execution environment that may be advantageously associated with a class, may be separately **listed in the Affinity Resource Table** and used in connection with the techniques disclosed herein. [emphasis added]

Breslau does not expressly disclose (a) a loader is used to determine availability; (b) that the system is a virtual subsystem; (c) creation of a native executable according to a particular user; (d) the native executable is utilized to provide improved performance of the virtual subsystem; or (e) the runtime logged information includes at least a set of information related to a particular user to create a native executable according to the particular user.

However, in an analogous environment, Armstrong teaches (a) and (b): *a loader to determine availability of a specialized image that is associated with an operating environment of the virtual subsystem*. See the "Control" block in the figure at the bottom of page 3; also see the top of page 4:

> When a method is invoked, the native machine-code version is used, if it exists.

Also in an analogous environment, Spyker teaches (c) that a user can control the creation of a runtime image. See Spyker column 14 lines 44-46 and 52-55:

> The process begins at Block 700, when **a user requests to execute an application**, and takes place on the client machine (as noted at element 710). [emphasis added]

Spyker further teaches (d): generating native executables on a virtual subsystem for improving the performance of the subsystem. See column 1, lines 22-27 and lines 37-44:

> Java attains its portability through use of a specially-designed virtual machine ("VM"). This virtual machine is also referred to as a "Java Virtual Machine", or "JVM". The virtual machine enables isolating the details of the underlying hardware from the compiler used to **compile** the Java programming instructions. [emphasis added]

Note that execution of an application necessarily requires the creation of a runtime image. It is the runtime image that is created, placed in memory, and executed.

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to use Armstrong's loader with Breslau's environment log. One would be motivated to use a loader that checks for specialized images in order to optimize execution time (Armstrong page 4 paragraph 2). It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Spyker's virtual subsystem with Breslau's environment. One of ordinary skill would have been motivated to allow a programming image to be utilized in a number of different environments that support a virtual subsystem. One of ordinary skill would have also been motivated to enable the creation of an executable runtime image according to a user in order to prepare the image for loading and execution at the user's discretion.

None of the above references expressly teaches feature (e) the runtime logged information includes at least a set of information related to a particular user to create a native executable according to the particular user. However, Knight teaches collecting runtime feedback associated with a particular user. See column 6 lines 22-27:

> As an application is being used by the developer, as generally shown by numeral 11, all **object identifiers** corresponding to IWindow objects within the application being changed, **affected by or interacted with in some way by the developer/user, are logged in object display area 15** of the screen of setup tool 10. [emphasis added]

Note that this logging of information is provided in the general context of program instrumentation (i.e. "logging") for the purpose of providing enhancements to software. E.g. see column 1 lines 30-36:

> Understanding what is important to users of a particular software application product has always been a challenge for software application developers. Typically, **enhancements to a product release** are based on feedback from surveys, forums or from user design centre sessions in which participants are given a set of tasks to accomplish while using the application. [emphasis added]

Knight's runtime logged feedback is used to create an enhanced software application (i.e. executable) that compliments the automatic compilation of executables disclosed by Breslau (see Breslau column 3 lines 10-14). It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Knight's teaching of user feedback with Breslau's native executable in order to overcome the problems with "compile-time" instrumentation (see Knight column 2 lines 12-22).

In regard to Claim 2, Breslau teaches that the native executable is selected for execution by the virtual subsystem by matching a current environment setting with the logged information (Column 8, lines 22-27).

In regard to Claim 5, Breslau teaches a local data log (Figure 4, item 135).

In regard to Claim 6, Breslau teaches a data log stores 1 though N environment parameter descriptions associated with 1 to N encountered images, wherein N is an integer (Figure 1A).

In regard to Claim 7, Spyker teaches a virtual machine as a virtual subsystem which uses an intermediate code image (Figure 1, lines 22-27 and lines 39-44).

In regard to Claim 8, Spyker teaches a Just-In-Time compilation (Column 1, lines 39-44).

In regard to Claim 9, Spyker teaches that the virtual subsystem generates native platform code (Column 1, lines 39-44).

In regard to Claim 10, Spyker teaches installing or running a generic code image by converting it into a native executable (Column 1, lines 39-44).

In regard to claim 11, Spyker teaches creating a runtime image according to a method of invocation (column 14 lines 48-52).

In regard to Claim 12, Spyker teaches generating a native code image using the virtual machine (Column 1, lines 39-44).

In regard to Claim 13, Breslau teaches am image processor for processing feedback and generating a native executable (Figure 3).

In regard to Claim 14, Breslau teaches that the image processor comprises a compiler (Figure 3, item 59).

In regard to Claim 15, Breslau teaches an image-processing tool to read the logged information and associate one or more environmental settings with one or more related images encountered during virtual subsystem execution (Column 9, lines 43-48).

In regard to Claim 16, Breslau teaches logged information relating to an operating system version and processor type (Figure 2).

In regard to Claim 17, Breslau teaches a system identifier to match parameters with native code (Figure 2, items "SYS A", "SYS B", and "SYS C").

In regard to Claim 19, Breslau teaches a medium (Figure 4) for carrying out said execution of the system in Claim 1.

Claim 30 corresponds with Claim 1, and Claim 30 is rejected for the same reasons as Claim 1, where a signal is an inherent aspect of communication in a data processing system. Spyker's generic image (Java Bytecode) is predetermined to be incompatible with the operating environment of the virtual system, since bytecode is an intermediate code format that is not directly executable. All further limitations have been addressed in the above rejection of claim 1.

In regard to Claim 31, Spyker teaches that this signal is communicated over a network (Figure 5A, items 506 and 507).

- Claims 3 and 4 are rejected under 35 U.S.C. 103(a) as being unpatentable over Breslau, Spyker, Armstrong, and Knight as applied in the above rejection of claims 1, 2, 5-10, 12-17, 19, 30, and 31, and further in view of prior art of record U.S. Patent Number 6,721,946 to Fogarty et al. (hereinafter "Fogarty").

In regard to Claim 3, Breslau and Spyker teach the method of Claim 1, but do not teach an image repository to store 1 through N specialized native images, wherein N is a positive integer. Fogarty, however, does teach an image repository for holding a plurality of software images (Figure 2, item 212). Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to build the system of Claim 1, further storing the images in an image repository, since this allows the images to be centrally accessed from one location.

In regard to Claim 4, Fogarty teaches that the image database is a local or remote database (Figure 3, item 212).

- Claim 18 is rejected under 35 U.S.C. 103(a) as being unpatentable over Breslau, Spyker, Armstrong, and Knight as applied in the above rejection of claims 1, 2, 5-10, 12-17, 19, 30, and 31, and further in view of prior art of record U.S. Patent Number 6,253,368 to Nelin et al. (hereinafter "Nelin").

In regard to Claim 18, Breslau and Spyker teach the system of Claim 16, but neither teaches that the developer parameters describe at least one of debug options, compiler switch settings and information relating to preferences of a user. Nelin, however, does teach storing development parameters that deal with user preferences of debug options (Column 15, lines 40-47). Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to build the system of Claim 16, as taught by Breslau and Spyker, where the developer parameters describe at least one of debug options, compiler switch settings and information relating to preferences of a user, as taught by Nelin, since these options are also a field that helps to profile the settings and preferences of a computer system and a user.

- Claims 20-22 and 27-29 are rejected under 35 U.S.C. 103(a) as being unpatentable over prior art of record U.S. Patent Number 6,158,049 to Goodwin et al. (hereinafter "Goodwin") in view of U.S. Patent No. 6,126,330 to Knight (hereinafter "Knight").

In regard to Claim 20, Goodwin discloses: determining a first code image associated with a possible runtime environment (Figure 1, item 105 – the determined first code image is the

instrumented object code); executing the first code image in an unmodified form in the runtime environment (Figure 2, item 151); and generating runtime feedback associated with the first code image to adjust a subsequent code image according to the runtime environment (Figure 2, items 152 and 107). Goodwin further discloses feedback that includes a set of information to create a code image (See column 16 lines 14-16 – in particular "profile data files"). Goodwin further discloses creation of an image according to a particular user (column 3 line 53 – column 4 line 60 generally describes the process of creation of an optimized executable image from the perspective of a particular user that enters commands – e.g. column 3 lines 53-54 "command from the user"). Goodwin does not expressly disclose runtime feedback associated with a particular user. However, Knight teaches collecting runtime feedback associated with a particular user (See column 6 lines 22-27: "As an application is being used by the developer, as generally shown by numeral 11, all object identifiers corresponding to IWindow objects within the application being changed, affected by or interacted with in some way by the developer /user, are logged in object display area 15 of the screen of setup tool 10"). It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Knight's teaching of user feedback with Goodwin's code image in order to overcome the problems with "compile-time" instrumentation (see Knight column 2 lines 12-22).

In regard to Claim 21, Goodwin teaches generating a specialized executable from the subsequent code image (Column 4, lines 57-60).

In regard to Claim 22, Goodwin teaches storing the application images in a database (Figure 1, item 107).

In regard to Claim 27, Goodwin teaches at least: organizing data and methods in the first image to optimize the images based on profile data (Column 2, lines 63-67).

In regard to Claims 28 and 29, these are system Claims that correspond with method Claims 20 and 21, and are rejected for the same reasons as Claim 20 and 21 respectively, where Goodwin teaches a system for carrying out said method of Claims 20 and 21 (Figure 1).

- Claims 23 and 24 is rejected under 35 U.S.C. 103(a) as being unpatentable over Goodwin and Knight as applied in the above rejection of claim 21, further in view of "Compilers: Principles, Techniques, and Tools" by Aho et al. (hereinafter "Aho").

In regard to Claim 23, Goodwin teaches processing a generic image using standard compilation techniques (Figure 1, item 102). Goodwin and Knight do not expressly teach intermediate language. However, in an analogous environment, Aho teaches processing intermediate language code utilizing standard compilation techniques. See Figure 1.9 on page 10; also pages 12-14. It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Aho's intermediate language with Goodwin's compiler. One of ordinary skill would have been motivated to use a language that is easily translated into a target program (see Aho, last full paragraph on page 12).

In regard to Claim 24, Goodwin teaches the method of Claim 23, but does not teach logging operating environment information during processing of the generic image. Knight, however, does teach logging environment variables of a computer system to compile a generic image (see column 6 lines 22-27). Therefore, it would have been obvious to one of ordinary skill

in the art at the time of the invention to perform the method of Claim 23, as taught by Goodwin, where the method includes logging operating environment information during processing of the generic image, as taught by Knight, since this allows customization of the image to suit the environment.

- Claims 25 and 26 are rejected under 35 U.S.C. 103(a) as being unpatentable over Goodwin, Knight, and Aho as applied to the rejection of claim 23 above, and further in view of Breslau. In regard to Claim 25, Goodwin teaches the method of Claim 23, and Knight teaches product enhancement, but neither teaches building the specialized executable to suit the environment. Breslau, however, does teach generating an environment specific executable (Column 1, lines 65-67). Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to perform the method of Claim 23, as taught by Goodwin, where the method includes building the specialized executable to suit the environment, as taught by Breslau, since this allows customization of the executable to suit the environment.

In regard to Claim 26, Breslau teaches selecting the specialized executable by matching a current environment setting with the logged environment information (Column 8, lines 22-27).

- Claim 32 is rejected under 35 U.S.C. 103(a) as being unpatentable over Breslau in view of Spyker, Knight, and Nelin.

In regard to Claim 32, Breslau teaches a first data field having parameters relating to at least one of an operating system version (Figure 2, item "OS" in SET Table) and a third data field having a set of information to create an executable image (Figure 3, item 59). Breslau does not teach a second data field having at least one of a developer parameter, a domain flag, a security information field, and a binding information field. Breslau also does not teach the creation of an executable image according to a particular user.

Nelin, however, does teach a developer parameter field for debugging programs (Column 15, lines 40-47). Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to construct a data structure containing a first data field having parameters relating to at least one of an operating system version and a third data field having a profile information field associated with the operating environment of a virtual system, as taught by Breslau, where the structure also contains a second data field having a developer parameter, as taught by Nelin, since a developer parameter is also a field that helps to profile the settings and preferences of a computer system and a user. Also, Spyker teaches creation of a runtime image according to a user (Spyker column 14 lines 44-46), and Knight teaches collecting runtime feedback associated with a particular user (See column 6 lines 22-27) as addressed in the above rejection of claim 1.

- Claim 33 is rejected under 35 U.S.C. 103(a) as being unpatentable over Breslau in view of prior art of record U.S. Patent 6,457,122 to Ramezani (hereinafter "Ramezani "), Knight, and Spyker.

In regard to Claim 33, Breslau teaches an execution engine that processes an image (Figure 3), the execution engine generating operating environment data while processing the image (Figure 2), and a specialized executable image generated at least in part from the operating environment data (Figure 3, item 59). Breslau does not teach that the specialized executable image stored in a repository of one or more other specialized executable images wherein the execution engine selects at least one specialized executable image from the repository if the at least one specialized image matches present operating environment data. Breslau also does not disclose the creation of an image according to a user.

Ramezani, however, does teach a specialized image repository (Column 4, lines 54-57); wherein the execution engine selects at least one specialized executable image from the repository if the at least one specialized image matches present operating environment data (Column 5, lines 11-12). Neither Breslau nor Ramezani teach that the image is an intermediate language image. Spyker, however, does teach processing an intermediate language image (Column 1, lines 37-44). Also, Spyker teaches creation of a runtime image according to a user (Spyker column 14 lines 44-46), and Knight teaches collecting runtime feedback associated with a particular user (See column 6 lines 22-27) as addressed in the above rejection of claim 1.

Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to build a system including an execution engine that processes an image, the execution engine generating operating environment data while processing the image, and a specialized executable image generated at least in part from the operating environment data, as taught by Breslau, where the specialized executable image stored in a repository of one or more other specialized executable images wherein the execution engine selects at least one specialized

executable image from the repository if the at least one specialized image matches present operating environment data, as taught by Ramezani, since this allows for a centralized storage location for all of the images, as well as an image that is designed specifically for a certain operating environment, further where the first image is an intermediate language image, as taught by Spyker, since this allows the image to be executed on any environment that can handle the intermediate language. Further, it would have been obvious to one of ordinary skill in the art at the time the invention was made to use Knight's teaching of user feedback with Breslau's native executable in order to overcome the problems with "compile-time" instrumentation (see Knight column 2 lines 12-22). One of ordinary skill in the art would be motivated to assess software in order to determine if it meets the requirements of the target system (See Ramezani's *Background* section in column 1 lines 23-24).

(10) Response to Argument

A. Rejection of Claims 1, 2, 5-17, and 19 Under 35 U.S.C. § 103(a)

In paragraph 2 on page 7 of the Brief, Appellant argues that the *Breslau* reference does not disclose information that is logged during runtime execution of an executable image. In response to applicant's arguments against the references individually, one cannot show nonobviousness by attacking references individually where the rejections are based on combinations of references. See *In re Keller*, 642 F.2d 413, 208 USPQ 871 (CCPA 1981); *In re Merck & Co.,* 800 F.2d 1091, 231 USPQ 375 (Fed. Cir. 1986). In this case, *Breslau* was not relied upon to disclose the feature of logging information during runtime execution. Rather,

Knight column 6 lines 22-27 was relied upon to disclose this limitation, as noted in the Final Rejection, which describes the runtime logging of information according to a particular user. This logging of information is provided in the general context of program instrumentation for the purpose of providing enhancements to software (e.g. see column 1 lines 30-36). Since *Breslau* was not relied upon to disclose this feature, Appellant's arguments are not persuasive.

Appellant further argues that the *Spyker* reference does not teach the creation of a runtime image according to a particular user. In the final rejection, this limitation was referenced with Spyker column 14 lines 44-46 and 52-55:

> The process begins at Block 700, when **a user requests to execute an application**, and takes place on the client machine (as noted at element 710).
> ...
> At Block 715, the process of **constructing the appropriate run-time environment for the application**, and starting the application executing in that environment, begins.
> [emphasis added]

These passages show that the creation of the runtime image occurs *according* to a particular user. The plain language of the claim simply calls for creation "according to the particular user." Reasonable broad interpretation allows *Spyker* to meet the plain language of the claim. Thus, Appellant's arguments are not persuasive.

At the top of page 8, Appellant argues that the *Spyker* and *Armstrong* references do not teach "information related to the user is logged during execution of the application." However, *Spyker* and *Armstrong* were not relied upon to teach this limitation. Rather, *Knight* (see column 6 lines 22-27) was provided to teach this limitation. Therefore, this argument is not persuasive.

In the middle of page 8, Appellant argues that the *Knight* reference does not teach the alleged "deficiencies" of the other references, including logged runtime information. *Knight*

column 6 lines 22-27 was cited to teach the logging of runtime information related to a particular developer/user:

> As an application is being used by the developer, as generally shown by numeral 11, all **object identifiers** corresponding to IWindow objects within the application being **changed, affected by or interacted with in some way by the developer/user, are logged** in object display area 15 of the screen of setup tool 10. [emphasis added]

This passage shows that interactions (i.e. "runtime information") related to a particular developer/user are logged. As noted above, this logging of information is provided in the general context of program instrumentation for the purpose of providing enhancements to software (e.g. see column 1 lines 30-36). As such, Knight's runtime logged feedback is used to create an enhanced software application (i.e. executable) that compliments the automatic compilation of executables disclosed by Breslau (see Breslau column 3 lines 10-14).

Appellant notes that *Knight* teaches the use of an input file to identify information for logging, but does not teach an input file containing runtime logged feedback (see middle of page 8 of Appellant's Brief). However, Knight's input file is not relied upon to teach runtime logging. In fact, *Knight* column 6 lines 22-27 is relied upon for the expressed teaching of runtime logging:

> **As an application is being used** by the developer, as generally shown by numeral 11, all **object identifiers** corresponding to IWindow objects within the application being changed, **affected by or interacted with in some way by the developer/user, are logged in object display area 15** of the screen of setup tool 10. [emphasis added]

This passage clearly shows "runtime logged information related to a particular user." Therefore, Appellant's arguments are not persuasive.

B. Rejection of Claims 30 and 31 Under 35 U.S.C. § 103(a)

At the top of page 9 of the Brief, Appellant argues that the prior art of record does not teach or suggest the invention for the same reasons as presented for claim 1. Likewise, these arguments are not persuasive for the reasons presented above.

C. Rejection of Claims 3 and 4 Under 35 U.S.C. § 103(a)

At the bottom of page 9 of the Brief, Appellant argues that the prior art of record does not teach or suggest the invention for the same reasons as presented for claim 1. Likewise, these arguments are not persuasive for the reasons presented above.

D. Rejection of Claims 30 and 31 Under 35 U.S.C. § 103(a)

On page 10 of the Brief, Appellant argues that the prior art of record does not teach or suggest the invention for the same reasons as presented for claim 1. Likewise, these arguments are not persuasive for the reasons presented above.

E. Rejection of Claims 20-22 and 27-29 Under 35 U.S.C. § 103(a)

On page 11 of the Brief, Appellant argues that the *Goodwin* and *Knight* references do not teach "code image runtime feedback related to a particular user" or creation of "a native executable image according to a particular user." *Goodwin* Figure 2, items 152 and 107 were cited to disclose the creation of a native executable image using runtime feedback. As presented in the above arguments regarding the rejection of claim 1, *Knight* column 6 lines 22-27 was cited

to teach the logging of runtime information related to a particular user. Likewise, these arguments are not persuasive for the same reasons presented above.

F. Rejection of Claims 23 and 24 Under 35 U.S.C. § 103(a)

On page 12 of the Brief, Appellant argues that the prior art of record does not teach or suggest the invention for the same reasons as presented for claim 20. Likewise, these arguments are not persuasive for the reasons presented above.

G. Rejection of Claims 25 and 26 Under 35 U.S.C. § 103(a)

On page 12 of the Brief, Appellant argues that the prior art of record does not teach or suggest the invention for the same reasons as presented for claims 21 and 23. Likewise, these arguments are not persuasive for the reasons presented above.

H. Rejection of Claim 32 Under 35 U.S.C. § 103(a)

On page 13 of the Brief, Appellant argues that the prior art of record does not teach or suggest the invention for the same reasons as presented for claims 1, 20, 28 and 30. Likewise, these arguments are not persuasive for the reasons presented above.

J. Rejection of Claim 33 Under 35 U.S.C. § 103(a)

On page 13 of the Brief, Appellant argues that the prior art of record does not teach or suggest the invention for the same reasons as presented for claim 1. Likewise, these arguments are not persuasive for the reasons presented above.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

/J. Derek Rutten/
Patent Examiner, Art Unit 2192

Conferees:

/Tuan Q. Dam/
Tuan Q. Dam
Supervisory Patent Examiner, Art Unit 2192

Eddie Lee
SPE, Technology Center 2100
/Eddie C Lee/

Supervisory Patent Examiner, TC 2100